Microprocessors Project

Mohamed Hossam 49-19321

Ismaiel Amr 49-4522

Mohamed Hatem 49-93983

Omar Wael 49-4927

Mohamed Seif 49-4198

Code Development Steps:

We stared off by creating different data type for each part of the project such as: Instruction, Register, Reservation Station, Load Buffer and Store Buffers. The memory was implemented as an array of integer values.

At the start of the code, we created arrays for each type of reservation station, an array for the load buffer containing the load stations and an array for the store buffer containing its stations. We start filling these arrays by specific integer values depending on how many stations we want to use for our test cases. Instructions are read from a txt file then parsed using a parser into an array of instructions. The memory is an array filled with values from 0 to 51. Also, the time taken to execute each instruction type is determined by integer variables.

The execution is split into three methods. The issue method issues the next instruction if a reservation that the instruction needs is available. We do that by checking all the reservation stations and buffers to see if there is an empty place to issue the instruction.

The second method is the execute method. It checks all the reservation stations and buffers to see if there is a station which is ready to execute (i.e it is not waiting for any operands to come from previous instructions). If it is ready to execute, it will decrement a timer unique to each station which starts as the time set in the beginning to finish this instruction type until the timer reaches zero. This means that it is ready to write back.

The third method is the write back method. It checks if there are any stations that have a timer of zero to do the write back. If more than one has a timer of zero, we write back the instruction that came first. The other instruction waits till the next cycle to do its write back. In the write back, we calculate the value of the addition, subtraction, multiplication or division and write back to the register file or any station that needs the value coming out of this specific instruction. LDs also write back the value from the memory in this stage.

Those three methods are placed in the another method called next(). This method also increments the cycle time. The method is called over and over until everything finishes execution.

The code is developed using Java and Java Swing for the GUI.

Test Cases:

The first test case:

LD F6 50

LD F2 20

MUL F0 F2 F4

SUB F8 F6 F2

DIV F10 F0 F6

ADD F6 F8 F2

The second test case:

```
MUL F3 F1 F2
ADD F5 F3 F4
Add F7 F2 F6
ADD F10 F8 F9
MUL F11 F7 F10
ADD F5 F5 F11
```